```
1     #
2     #       Tuxedo configuration UBBCONFIG for the model TEST1
3     #
4
5     *RESOURCES
6     IPCKEY                  191785
7     MASTER                  site1
8     DOMAINID                        dom1
9     MAXACCESSERS    50
10    MAXSERVERS              50
11    MAXSERVICES             100
12    OPTIONS                 LAN
13    MODEL                   MP
14
15    *MACHINES
16    puce            LMID=site1
17                    TUXDIR="usr/tuxedo"
18                    APPDIR="/home/dia/tuxedo"
19                    TUXCONFIG="/home/dia/tuxedo/TUXCONFIG"
20                    ENVFILE="/home/dia/tuxedo/envfile_puce"
21                    ULOGPFX="/home/dia/tuxedo/ULOG"
22
23    trifide         LMID=site2
24                    TUXDIR="usr/tuxedo"
25                    APPDIR="/home/dia/tmp"
26                    TUXCONFIG="/home/dia/tmp/TUXCONFIG"
27                    ENVFILE="/home/dia/tmp/envfile_trifide"
28                    ULOGPFX="/home/dia/tmp/ULOG"
29
30    zig             LMID=site3
31                    TUXDIR="usr/tuxedo"
32                    APPDIR="/home/dia/tuxedo"
33                    TUXCONFIG="/home/dia/tuxedo/TUXCONFIG"
34                    ENVFILE="/home/dia/tuxedo/envfile_zig"
35                    ULOGPFX="/home/dia/tuxedo/ULOG"
36
37    orage           LMID=site4
38                    TUXDIR="usr/tuxedo"
39                    APPDIR="/home/dia/tuxedo"
40                    TUXCONFIG="/home/dia/tuxedo/TUXCONFIG"
41                    ENVFILE="/home/dia/tuxedo/envfile_orage"
42                    ULOGPFX="/home/dia/tuxedo/ULOG"
43
44
45
46    *GROUPS
47
48    DEFAULT:        TMSNAME=TMS     TMSCOUNT=2
49    GROUP1          LMID=site1
50                    GRPNO=1
51    GROUP2          LMID=site2
52                    GRPNO=2
53    GROUP4          LMID=site3
54                    GRPNO=3
55    GROUP3          LMID=site4
56                    GRPNO=4
57
58
59    *SERVERS
60    #
61    DEFAULT: RESTART=Y MAXGEN=5 REPLYQ=Y CLOPT="-A"
62
63    SRV1
64                    SRVGRP=GROUP1
65                    SRVID=100
66                    MIN=2   MAX=2
67                    RQADDR=QSRV1_1
68                    REPLYQ=Y
69                    CLOPT="-s SVC1_1 -s SVC1_2 -- "
70    SRV2
71                    SRVGRP=GROUP2
```

```
72                      SRVID=200
73                      MIN=2   MAX=2
74                      RQADDR=QSRV2_2
75                      REPLYQ=Y
76                      CLOPT="-s SVC2_1 -s SVC2_2 -- "
77      SRV4
78                      SRVGRP=GROUP4
79                      SRVID=300
80                      MIN=2   MAX=2
81                      RQADDR=QSRV4_3
82                      REPLYQ=Y
83                      CLOPT="-s SVC4_1 -s SVC4_2 -- "
84      SRV3
85                      SRVGRP=GROUP3
86                      SRVID=400
87                      MIN=2   MAX=2
88                      RQADDR=QSRV3_4
89                      REPLYQ=Y
90                      CLOPT="-s SVC3_1 -- "
91
92
93      *SERVICES
94      DEFAULT:        LOAD=50
95      SVC1_1
96      SVC1_2
97      SVC2_1
98      SVC2_2
99      SVC4_1
100     SVC4_2
101     SVC3_1
102
103
104
105     *NETWORK
106     site1
107     #       port number=60951 (ee17 hexa)
108     #       local address=81b683e0
109             NADDR="\\x0002ee1781b683e00000000000000000"
110             BRIDGE="/dev/xti/tcp"
111     #       port number=60952 (ee18 hexa)
112             NLSADDR="\\x0002ee1881b683e00000000000000000"
113     #
114     site2
115     #       port number=60951 (ee17 hexa)
116     #       local address=81b68387
117             NADDR="\\x0002ee1781b6838700000000000000000"
118             BRIDGE="/dev/xti/tcp"
119     #       port number=60952 (ee18 hexa)
120             NLSADDR="\\x0002ee1881b6838700000000000000000"
121     #
122     site3
123     #       port number=60951 (ee17 hexa)
124     #       local address=81b683e1
125             NADDR="\\x0002ee1781b683e10000000000000000"
126             BRIDGE="/dev/xti/tcp"
127     #       port number=60952 (ee18 hexa)
128             NLSADDR="\\x0002ee1881b683e10000000000000000"
129     #
130     site4
131     #       port number=60951 (ee17 hexa)
132     #       local address=81b6838b
133             NADDR="\\x0002ee1781b6838b0000000000000000"
134             BRIDGE="/dev/xti/tcp"
135     #       port number=60952 (ee18 hexa)
136             NLSADDR="\\x0002ee1881b6838b0000000000000000"
137     #
138
```

```
1
2     # @BULL_COPYRIGHT@
3     #
4     # HISTORY
5     # $Log: smtuxadmin.ksh,v $
6     # Revision 1.7  1996/02/12  11:40:49  odeadm
7     #      bci V1Set2C 23.01.96
8     #      [1996/01/23  14:31:07  dia]
9     #
10    # Revision 1.6  1995/12/20  14:26:59  odeadm
11    #      V1 Set2: Still troubles with smtuxadmin.ksh
12    #      [1995/12/11  11:56:55  odeadm]
13    #
14    #      07.12.95 V1Set2 first batch of corrections
15    #      [1995/12/07  17:22:57  odeadm]
16    #
17    #      *** empty log message ***
18    #      [1995/11/30  13:48:30  dia]
19    #
20    #      *** empty log message ***
21    #      [1995/11/30  13:48:30  dia]
22    #
23    # Revision 1.5  1995/10/13  11:52:51  odeadm
24    #      Servers TMS/Partitioned mach.
25    #      [1995/10/09  12:05:57  dia]
26    #
27    # Revision 1.4  1995/09/15  15:15:06  odeadm
28    #      Corrections MRs BUILD 3
29    #      [1995/09/07  15:45:27  dia]
30    #
31    # Revision 1.3  1995/08/24  13:38:03  odeadm
32    #      Build3
33    #      [1995/08/23  09:04:31  odeadm]
34    #
35    # Revision 1.2  1995/07/19  15:18:13  odeadm
36    #      Madison build M0.2
37    #      [1995/07/10  10:01:58  odeadm]
38    #
39    # $EndLog$
40    #! /bin/ksh
41    ConfDir=$WRAPPING_CONFIGURATION
42    Context=smtuxedo.ctx
43    Scanconf=$MADISON_VAR/surveyor/scanconf.tux
44    V5_to_V4='ROOTDIR=$TUXDIR; export ROOTDIR'
45    Set1_to_Set2='[ -z "$ADMIN" ] && export ADMIN="madison"'
46    cmd=$1; shift
47
48    set_environ() {
49          MASTER=""; APPDIR=""; ADMIN=""
50          filename=$ConfDir/$appname.tux
51    Env=`tuxgetenv -k -v APP_PW $filename << !
52    tuxgetenvp
53    !`
54          eval "$Env"; unset APP_PW
55          eval "$Set1_to_Set2"
56          if [ -n "$MASTER" -a -n "$APPDIR" ]
57          then
58                Env="$Env
59    $PW
60    $Set1_to_Set2
61    $V5_to_V4"'
62    LD_LIBRARY_PATH=$LIBPATH; export LD_LIBRARY_PATH;
63    cd $APPDIR
64    PATH=${PATH}:.:$APPDIR:$TUXDIR/bin; export PATH'
65                return 0
66          fi
67          exit 1
68    }
69
70    remote_cmd() {
71          prog="$Env
```

```
$cmd"'
status=$?
sleep 1
echo "\nexit $status"
'
#print -r "$prog" > prog
       rsh "$MASTER" -l "$ADMIN" "$prog" | awk '
              NR == 1 {line = $0}
    NR > 1  { print line; line = $0}
              END {if(sub("^exit ","", line)) exit line; exit -1 }'
}


get_tuxconfig() {
       if [ -s tuxconf.tmp.$appname ]
       then
              cat tuxconf.tmp.$appname
       else
              rm  -f tuxconf.tmp.*
              prog="$Env"'
$TUXDIR/bin/tmunloadcf
echo "\nexit $?"
'
#print -r "$prog" > prog
              rsh "$MASTER" -l "$ADMIN" "$prog" | tee tuxconf.tmp.$appname
       fi
get_tlistenlog
}

get_tlistenlog() {
       tllogfname=$ConfDir/tlistenlog.$appname.$machine
if [ -s $tllogfname ]
 then
   cat $tllogfname
 else # default value
   echo "TLLOG $machine $MADISON_TMP/tlisten.$appname.$machine.log" | tee $tllogfname
fi
   echo "\nexit $?"
}

get_tuxval() {
       get_tuxconfig | \
       sed -e "s/=/ /g" -e 's/"//g' -e 's/\\\\/0/g' | awk '
  BEGIN {
    tuxconfig_section["*RESOURCES"] = 1
    tuxconfig_section["*MACHINES"]  = 2
    tuxconfig_section["*GROUPS"]  = 3
    tuxconfig_section["*SERVERS"] = 4
    tuxconfig_section["*SERVICES"]  = 5
    tuxconfig_section["*ROUTING"] = 6
    tuxconfig_section["*NETWORK"] = 7
}
 NF == 1 {
    if ( $1 in tuxconfig_section ) {
    section = tuxconfig_section[$1]
    next
    }
  }
 section == 2 && $2 == "LMID" {  # MACHINES section
 if ( $3 == machine) {
   printf "uname=%s\n", $1
   mach_found=1
  }
  else { # reset mach_found for furtheur machines
   mach_found = 0
  }
  next
 }
section == 2 && $1=="APPDIR" && mach_found==1 {
              printf "appdir=%s\n", $2
    appdir = $2
```

```
143       next
144          }
145   section == 2 && $1=="TUXCONFIG" && mach_found == 1 {
146                 printf "tuxconfig=%s\n", $2
147       next
148          }
149   section == 2 && $1=="TUXDIR" && mach_found==1 {
150         printf "tuxdir=%s\n", $2
151     next
152         }
153   section == 2 && $1=="ROOTDIR" && mach_found==1 { # for V4
154                 printf "tuxdir=%s\n", $2
155       next
156          }
157   section == 2 && $1=="ULOGPFX" && mach_found==1 {
158       ulogpfx=1; printf "ulogpfx=%s\n", $2
159       next
160       }
161   section == 7 && NF == 1 {
162     if ( $1 == machine )
163       {mach_found = 1}
164     else { # reset mach_found for other machines
165       mach_found = 0
166       }
167     next
168     }
169   section == 7 && $1=="NLSADDR" && mach_found==1 {
170         printf "nlsaddr=%s\n", $2
171     next
172         }
173   section == 1 && $1 == "UID" {printf "uid=%s\n", $2 ;next }
174   section == 7 &&     $1=="BRIDGE" && mach_found==1 {
175                 printf "bridge=%s\n", $2 }
176     END { # not defined ulogpfx
177      if ( ulogpfx == 0 ) {
178       printf "ulogpfx=%s/ULOG\n", appdir }
179          } ' machine=$machine appname=$appname
180         lang=`sed -e "s/=/ /g" -e "s/'//g" -e "s/;/ /" $ConfDir/$appname.tux | awk '
181           $1 == "LANG" {printf "lang=", $2}' `
182       }
183
184   get_tllog() {
185   tllogfname="$ConfDir/tlistenlog.$appname.$machine"
186   if [ -f $tllogfname ]
187    then
188     tllog=`cat $tllogfname|awk '$1 == "TLLOG" && $2 == machine { print $3 }' machine=$m
      achine`
189    else
190      tllog="$MADISON_TMP/tlistenlog.$appname.$machine"
191      echo "TLLOG $machine $tllog" > $tllogfname
192   fi
193   }
194
195
196   case $cmd in
197         appli)
198                 ls -l $ConfDir 2> /dev/null | awk '
199                       sub(".tux$", "", $NF) {print $NF}'
200                 ;;
201         isexist)
202                 if [ -f $ConfDir/$1.tux ]
203                 then
204                       echo "Yes"
205                 else
206                       echo "No"
207                 fi
208                 ;;
209         setparam)
210                 [ ! -d $ConfDir ] && mkdir -p $ConfDir
211                 if [ -n "$2" ]
212                 then
```

```
213                     filename=$ConfDir/$2.tux
214                     while [ $# -gt 0 ]
215                     do
216                             echo "$1=\"$2\"; export $1"
217                             shift 2
218                     done > $filename
219             fi
220             ;;
221     discover)
222             [ -z "$1" ] && exit 1
223             filename=$ConfDir/$1.tux; shift
224             if [ -f $filename ]
225             then
226     #               sed -e 's/:/@@@/g' -e 's/#.*//' -e 's/ *; */"/g' $filename/ |
    awk '
227                     sed  -e 's/#.*//' -e 's/ *; */"/g' -e 's/:/#!:/g' $filename/
    | awk '
228                     BEGIN { field = "#promptW:promptP:promptPO:promptS:promptA:pr
    omptM:promptC:promptR:promptF"; value="::::::::" }
229                     /\=/ {
230                             for (i=1; i<= NF; i++) {
231                                 if(sub("=$", "", $i)) {
232                                    separator = ":"
233                                    field = field separator $i
234                                    value = value separator $(i+1)
235                                 }
236                             }
237                     }
238                     END {
239                             print field; print value
240                     }' FS='"'
241             else
242                     print '#\n'
243             fi
244             ;;
245     delappname)
246             if [ -n "$2" ]
247             then
248                     filename=$ConfDir/$2.tux
249                     if [ -f $filename ] && grep -q "$1=['\"]*$2" $filename
250                     then
251                             rm -f $filename ${filename}p
252                     else
253                             echo 'The file does not exist'
254                             echo '          or'
255                             echo 'The file is not an environment file'
256                             exit 1
257                     fi
258             fi
259             ;;
260     select)
261             if [ -n "$2" ]
262             then
263                     echo "$1='$2'; export $1" > "$Context"
264             fi
265             ;;
266     deselect)
267             rm -f "$Context"
268             ;;
269     selected)
270             APPNAME=""
271             [ -f $Context ] && . ./$Context
272             echo "$1$APPNAME"
273             ;;
274     isselected)
275             rm -f tuxconf.tmp.*
276             [ -f $Context ] && fgrep -q "APPNAME=" $Context && shift
277             echo $1
278             ;;
279     loadcf)
280             appname=$1
```

loop

```
281       boucle_status=0
282               cmd="\$TUXDIR/bin/tmloadcf -y $2 $3"
283               set_environ
284      echo "---- Loading Configuration Binary File ---"
285               remote_cmd
286       status=$?
287       if [ $status -ne 0 ]
288        then
289         exit $status
290        else
291   # maj fichier $Scanconf.tux machines
292         prog="$Env"'
293         $TUXDIR/bin/tmunloadcf
294         echo "\nexit $?"
295         '
296         #print -r "$prog" > prog
297         rsh "$MASTER" -l "$ADMIN" "$prog" > tuxconf.tmp.$appname
298         list_lmids=`cat tuxconf.tmp.$appname | sed -e "s/=/ /g" -e 's/"//g' -e "s/\*//
    "  | awk '
299           {line = $0}
300           $2 == "LMID" && machine == 1 {lmids = lmids $3 " "; next}
301           $1 == "GROUPS" && $2 == "" { machine=0; next}
302           $1 == "MACHINES" && $2 == "" { machine = 1; next}
303           END {if(sub("^exit ","", line)) {
304              print lmids
305              exit line}
306              exit -1 }'`
307         for machine in $list_lmids
308           do
309           echo "---- Updating $Scanconf on $machine ----\n"
310           get_tuxval > "appname.tux"
311           . ./appname.tux
312           log_prefix=`echo $ulogpfx | sed -e 's./. .g' | awk '
313              {print $NF} '`
314           log_dir=`echo $ulogpfx | sed -e 's./. .g' | awk '
315             {for (i=1; i< NF; i++) {
316               tempo = tempo "/" $i }}
317             END { print tempo}'`
318   #Build the 3 lines of $Scanconf for the application
319           prog="
320   [ -x $MADISON_BIN/security/updscantux ] &&
321   $MADISON_BIN/security/updscantux $appname $log_dir $log_prefix
322   echo \"\\nexit \$?\""
323            rsh "$uname" -l madison "$prog"  | awk '
324               NR == 1 {line = $0}
325               NR > 1  { print line; line = $0}
326               END {if(sub("^exit ","", line)) exit line; exit -1 }'
327          boucle_status=`expr $boucle_status + $? `
328        done
329       fi
330       exit $boucle_status
331               ;;
332         apppwd)
333               filename=$ConfDir/$1.tuxp
334               echo "Enter Application Password: \c"
335               OLDCONFIG=`stty -g`
336               stty -echo
337               read APP_PW
338               echo "\nRe-enter Application Password: \c"
339               read APP_PW_1
340               stty $OLDCONFIG
341               if [ "$APP_PW" != "$APP_PW_1" ]
342               then
343                     echo "\n\nPassword mismatch!"
344                     echo "Enter any character to exit and retry"
345                     read
346               else
347         #           PWencode "APP_PW=\"$APP_PW\"; export APP_PW" > $filename
348           #             APP_PW=`echo $APP_PW | sed -e "s/'/'\"'\"'/g"`
349         #           PWencode "APP_PW='$APP_PW'; export APP_PW" > $filename
350   tuxgetenv -s > $filename << !
```

```
tuxgetenvp
$APP_PW
!
                fi
                ;;
        chksyntx)
                appname=$1
                cmd="\$TUXDIR/bin/tmloadcf -n $2"
                set_environ
                remote_cmd
                exit $?
                ;;
        dispIpc)
                appname=$1
                cmd="\$TUXDIR/bin/tmloadcf -c $2"
                set_environ
                remote_cmd
                exit $?
        ;;
        machine_network)
                appname=$1
                set_environ
                get_tuxconfig | \
                sed -e "s/=/ /g" -e 's/"//g' -e 's/\\//' -e "s/\*//"  | awk '
                        BEGIN { network=0 }
                        {line = $0}
                        NF == 1 { if (network == 1) print $1}
                        $1 == "NETWORK" { network = 1}
                        END {if(sub("^exit ","", line)) exit line; exit -1 }'
                exit $?
                ;;

        machine_machines)
                appname=$1
                set_environ
                get_tuxconfig | \
                sed -e "s/=/ /g" -e 's/"//g' -e 's/\\//' -e "s/\*//"  | awk '
                        BEGIN { machine=0 }
                        {line = $0}
                        $2 == "LMID" { if(machine == 1) print $3}
                        $1 == "GROUPS" { if( $2 == "" ) machine=0}
                        $1 == "MACHINES" { if( $2 == "" ) machine = 1}
                        END {if(sub("^exit ","", line)) exit line; exit -1 }'
                exit $?
                ;;
        group)
                appname=$1
                set_environ
                get_tuxconfig | \
                sed -e "s/=/ /g" -e 's/"//g' -e 's/\\//' -e "s/\*//"  | awk '
                        BEGIN { group=0 }
                        {line = $0}
                        $1 == "SERVERS" { group=0 }
                        $1 == "GROUPS"  { if($2 == "") group=1}
                        $2 == "LMID" && $4 == "GRPNO" { if(group) print $1}
                        END {if(sub("^exit ","", line)) exit line; exit -1 }'
                exit $?
                ;;
        svrname)
                appname=$1
                set_environ
                get_tuxconfig | \
                sed -e "s/=/ /g" -e 's/"//g' -e 's/\\//' -e "s/\*//"  | awk '
                        BEGIN { group=server=nb_of_distinct_svr_name=0 }
                        {line = $0}
                        $1 == "TMSNAME" { if ( group == 1) {
                            trouve = 0
                            if (nb_of_distinct_svr_name == 0) {
                               nb_of_distinct_svr_name=1
                               svr_names[nb_of_distinct_svr_name] = $2
                               print $2
```

```
422                     } else {
423                        for (j=1; j<= nb_of_distinct_svr_name; j++) {
424                              if ( $2 == svr_names[j] ) {
425                                    trouve=1
426                              }
427                        }
428                        if (trouve == 0) {
429                              nb_of_distinct_svr_name += 1
430                              svr_names[nb_of_distinct_svr_name] = $2
431                              print $2
432                           }
433                        }
434                     }
435                  }
436                  $1 == "SERVERS" { if ($2 == "") {
437                        server=1
438                        group=0 }
439                  }
440                  $1 == "SERVICES" { if ($2== "") server=0}
441                  $1 == "GROUPS"   { if ($2 == "") group=1}
442                  $2 == "SRVGRP" {
443                      if((server == 1) && ( $4 == "SRVID")) {
444                        trouve = 0
445                        if (nb_of_distinct_svr_name == 0) {
446                            nb_of_distinct_svr_name = 1
447                            svr_names[nb_of_distinct_svr_name] = $1
448                            print $1
449                        } else {
450                            for(j=1; j<= nb_of_distinct_svr_name; j++) {
451                              if ( $1 == svr_names[j] ) {
452                                    trouve=1
453                              }
454                            }
455                            if(trouve == 0) {
456                              nb_of_distinct_svr_name += 1
457                              svr_names[nb_of_distinct_svr_name] = $1
458                              print $1
459                            }
460                        }
461                     }
462                  }
463                  END {if(sub("^exit ","", line)) exit line; exit -1 }'
464         exit $?
465         ;;
466   svrseq)
467         appname=$1
468         set_environ
469         get_tuxconfig | \
470         sed -e "s/=/ /g" -e 's/"//g' -e 's/\\//' -e "s/\*//"  | awk '
471               BEGIN { server=0; nb_of_distinct_svr_seq=0 }
472               {line = $0}
473               $1 == "SEQUENCE" && server == 1 {
474                  trouve = 0
475                  if (nb_of_distinct_svr_seq == 0) {
476                     nb_of_distinct_svr_seq=1
477                     svr_seqs[nb_of_distinct_svr_seq] = $2
478                     print $2
479                  } else {
480                     for (j=1; j<= nb_of_distinct_svr_seq; j++) {
481                           if ( $2 == svr_seqs[j] ) {
482                                 trouve=1
483                           }
484                     }
485                     if (trouve == 0) {
486                           nb_of_distinct_svr_seq += 1
487                           svr_seqs[nb_of_distinct_svr_seq] = $2
488                           print $2
489                     }
490                  }
491               }
492               $1 == "SERVERS" {  if($2 == "") server=1)
```

```
493                 $1 == "SERVICES" { if($2 == "") server=0}
494                 END {if(sub("^exit ","", line)) exit line; exit -1 }'
495         exit $?
496         ;;
497 svrId)
498         appname=$1
499         set_environ
500         get_tuxconfig | \
501         sed -e "s/=/ /g" -e 's/"//g' -e 's/\\//' -e "s/\*//"  | awk '
502               BEGIN { server=0; nb_of_distinct_svr_Id=0 }
503               {line = $0}
504               $2 == "SRVGRP" && $4 == "SRVID" &&  server == 1 {
505                   trouve = 0
506                   if (nb_of_distinct_svr_Id == 0) {
507                      nb_of_distinct_svr_Id=1
508                      svr_Ids[nb_of_distinct_svr_Id] = $5
509                      print $5
510                   } else {
511                      for (j=1; j<= nb_of_distinct_svr_Id; j++) {
512                            if ( $5 == svr_Ids[j] ) {
513                                  trouve=1
514                            }
515                      }
516                      if (trouve == 0) {
517                            nb_of_distinct_svr_Id += 1
518                            svr_Ids[nb_of_distinct_svr_Id] = $5
519                            print $5
520                      }
521                   }
522               }
523               $1 == "SERVERS" {  if($2 == "") server=1}
524               $1 == "SERVICES" { if($2 == "") server=0}
525               END {if(sub("^exit ","", line)) exit line; exit -1 }'
526         exit $?
527         ;;
528 discover_conf)
529         machine=$2
530         appname=$1
531         set_environ
532         get_tuxconfig | \
533         sed -e "s/=/ /g" -e 's/"//g' -e 's/\\\\/0/' -e "s/\*//"  | awk '
534               BEGIN {field = "#" }
535               {line = $0}
536               $1 == "UID" {
537                     field = field separator $1
538                     value = value separator $2
539                     separator = ":"
540               }
541               $1 == "GID" {
542                     field = field separator $1
543                     value = value separator $2
544                     separator = ":"
545               }
546
547               $1 == "BRIDGE" && network == 1 && mach_found == 1 {
548                     field = field separator $1
549                     value = value separator $2
550               }
551                    $1 == "NLSADDR" && network == 1 && mach_found == 1 {
552                           field = field separator $1
553                           value = value separator $2
554                     network = 0
555                     mach_found = 0
556                    }
557                    $1 == "TLLOG" && $2 == machine {
558                           field = field separator $1
559                     value = value separator $3
560                    }
561
562               $1 == machine {mach_found = 1}
563               $1 == "NETWORK"  { network = 1}
```

ANNEXE 2
APPENDIX

```
564                     END {
565                           print field; print value
566                           if(sub("^exit ","", line)) exit line; exit -1
567                     }' "machine=$machine"
568               exit $?
569               ;;
570         chglisten)
571                  appname=$1
572                  machine=$2
573                  shift 2
574                  if [ $# -gt 0 ]
575                   then
576                     echo "TLLOG $machine $1" > $ConfDir/tlistenlog.$appname.$machine
577                  fi
578                  exit $?
579               ;;
580         chklistscript)
581               appname=$1
582                  machine=$2
583                  set_environ
584                  get_tuxval > "appname.tux"
585                  get_tllog
586                  . ./appname.tux
587                  prog="
588                  if [ -f $appdir/tlisten.$appname.$machine ]
589                   then
590                     cat $appdir/tlisten.$appname.$machine
591                     echo \"\\nexit 0\"
592                   else
593                     echo \"\\nexit 1\"
594                  fi"
595                  if [ -z "$uname" ]
596                    then
597                      print "Host $machine not found"
598                      exit 1
599                  fi
600                  rm -f tlscript.$appname.$machine
601                  rsh "$uname" -l "$ADMIN" "$prog" | tee tlscript.$appname.$machine > /
    dev/null
602                  [ $? -ne 0 ] && exit 1
603                  [ -s tlscript.$appname.$machine ] && cat tlscript.$appname.$machine |
    awk '
604                    END { if ( $2 == "1" ) exit -1} '
605                  [ $? -eq -1 ] && exit 1
606                  [ -s tlscript.$appname.$machine ] && cat tlscript.$appname.$machine |
    \
607                  awk '
608                   $1 ~ "tlisten" {
609                       mismatch = 0
610                       fexec=sprintf("%s/bin/tlisten", tuxdir)
611                       if ($1 != fexec) {
612                          print "tlisten command full pathnames mismatch"
613                          printf "\tscript:\t%s\n", $1
614                          printf "\tconfig:\t%s\n", fexec
615                          mismatch +=1
616                          }
617                       for (i=2; i <= NF; i++) {
618                          if (( $i == "-d") && ($(i+1) != bridge)) {
619                             print "BRIDGE values mismatch"
620                             printf "\tscript:\t%s\n",$(i+1)
621                             printf "\tconfig:\t%s\n",bridge
622                             mismatch +=1
623                             }
624                           if (( $i == "-l") && ($(i+1) !=nlsaddr)) {
625                             print "NLSADDR values mismatch"
626                             printf "\tscript:\t%s\n",$(i+1)
627                             printf "\tconfig:\t%s\n", nlsaddr
628                             mismatch +=1
629                             }
630                           if (( $i == "-u") && ($(i+1) != uid)) {
631                             print "UID values mismatch"
```

ANNEXE 2
APPENDIX

```
632                             printf "\tscript:\t%s\n", $(i+1)
633                             printf "\tconfig:\t%s\n",uid
634                             mismatch +=1
635                             }
636                           if (( $i == "-L") && ($(i+1) !=tllog)) {
637                             print "LOGFILE values mismatch"
638                             printf "\tscript:\t%s\n", $(i+1)
639                             printf "\tconfig:\t%s\n", tllog
640                             mismatch +=1
641                             }
642                           }}
643                   END {
644                      if ( mismatch == 0 )
645                       printf "Script File is up-to-date for %s\n",machine
646                      else
647                       printf "\nScript File is NOT up-to-date for %s\n",machine
648                      } ' tllog=$tllog machine=$machine bridge=$bridge \
649                           nlsaddr=$nlsaddr uid=$uid tuxdir=$tuxdir
650                exit $?
651                ;;
652        startlistproc)
653                appname=$1; shift
654                list="$*"
655                set_environ
656                boucle_status=0                    (boucle = loop)
657                exit_status=0
658                for machine in $list
659                do
660                   echo "\n------- Machine: $machine -------\n"
661                   get_tuxval > "appname.tux"
662                      get_tllog
663                      . ./appname.tux
664                      prog1="
665                      TUXDIR=$tuxdir; export TUXDIR
666                   ROOTDIR=$tuxdir; export ROOTDIR  # V4
667                      APPDIR=$appdir; export APPDIR
668                      TUXCONFIG=$tuxconfig; export TUXCONFIG
669                      PATH=${PATH}:\$TUXDIR/bin:\$APPDIR; export PATH
670                      LANG=$lang; export LANG
671                      LIBPATH=${LIBPATH}:$tuxdir/lib; export LIBPATH
672                      COLUMNS=200; export COLUMNS
673                      ps -eF '%u %p %a' | awk '\$3 ~ \"tlisten\" && \$0 ~ \"$nlsaddr\" {
    exit 1}'
674                      if [ \$? = 1 ]
675                        then
676                           echo \"Listener already running on $machine\"
677                           echo exit 0
678                         exit 0
79                              fi
680                      if [ -f $appdir/tlisten.$appname.$machine ]
681                      then
682                           . $appdir/tlisten.$appname.$machine
683                         ps -eF '%u %p %a' | awk '\$3 ~ \"tlisten\" && \$0 ~ \"$nls
    addr\" {exit 1}'
684                         if [ \$? = 1 ]
685                         then
686                             echo \"Listener started on $machine\"
687                             echo exit 0
688                         else
689                           echo \"Listener starting failed on $machine !!!\"
690                             echo exit 1
691                         fi
692                       else # create the script file & exec it
693                           echo \"$tuxdir/bin/tlisten -d $bridge -l $nlsaddr -u $uid -L
    $tllog\" > $appdir/tlisten.$appname.$machine
694                         chmod ug+x $appdir/tlisten.$appname.$machine
695                         . $appdir/tlisten.$appname.$machine
696                         ps -eF '%u %p %a' | awk '\$3 ~ \"tlisten\" && \$0 ~ \"$nlsadd
    r\" {exit 1}'
697                         if [ \$? = 1 ]
698                          then
```

```
699                             echo \"Listener started on $machine\"
700                             echo exit 0
701                           else
702                               echo \"Listener starting failed on $machine !!!\"
703                               echo exit 1
704                           fi
705                     fi"
706         #echo "$prog1" > prog1
707         if [ -z "$uname" ]
708            then
709               print "Host $machine not found"
710               exit 1
711         fi
712       rsh "$uname" -l "$ADMIN" "$prog1" | awk '
713             NR == 1 {line = $0}
714             NR > 1 { print line; line = $0 }
715             END {if(sub("^exit ","", line)) exit line; print line; exit -1}'
716       boucle_status=`expr $boucle_status \| $? `
717            done
718            exit $boucle_status
719      ;;
720    stoplistproc)
721       appname=$1; shift
722       list="$*"
723       set_environ
724       boucle_status=0
725       exit_status=0
726       for machine in $list
727          do
728          echo "\n------- Machine: $machine -------\n"
729            get_tuxval > "appname.tux"
730            . ./appname.tux
731            prog1="
732          COLUMNS=200; export COLUMNS
733          ps -eF '%u %p %a' | awk '\$3 ~ \"tlisten\" && \$0 ~ \"$nlsaddr\" {print \$
    2; exit 0 }' | read pid
734          if [ -n \"\$pid\" ]
735            then
736              kill -9 \$pid > /dev/null
737              status=\$?
738              if [ \$status -eq 0 ]
739              then
740                     echo \"Process \$pid killed on $machine\"
741                     echo exit 0
742              else
743                     echo \"Failed to stop listener on $machine!!!\"
744                     echo exit 1
745              fi
746          else
747              echo \"No Listener running on $machine\"
748              echo exit 1
749          fi"
750            if [ -z "$uname" ]
751               then
752                  print "Host $machine not found"
753                  exit 1
754            fi
755          rsh "$uname" -l "$ADMIN" "$prog1" | awk '
756             NR == 1 {line = $0}
757             NR > 1 { print line; line = $0 }
758             END {if(sub("^exit ","", line)) exit line; print line; exit -1}'
759          boucle_status=`expr $boucle_status \| $? `
760           done
761       exit $boucle_status
762      ;;
763
764         runninglist)
765          appname=$1
766          boucle_status=0
767          set_environ
768          list_lmids=`get_tuxconfig | \
```

```
769           sed -e "s/=/ /g" -e 's/"//g' -e 's/\\\\/0/' -e "s/\*//"  | awk '
770              BEGIN { network=0 }
771              {line = $0}
772              NF == 1 { if (network == 1) print $1}
773              $1 == "NETWORK" { network = 1}
774              END {if(sub("^exit ","", line)) exit line; exit -1 }' `
775           for machine in $list_lmids
776             do
777               get_tuxval > "appname.tux"
778               . ./appname.tux
779               prog1="
780               TUXDIR=$tuxdir; export TUXDIR
781               LIBPATH=${LIBPATH}:$tuxdir/lib; export LIBPATH
782               ROOTDIR=$tuxdir; export ROOTDIR  # V4
783               APPDIR=$appdir; export APPDIR
784               TUXCONFIG=$tuxconfig; export TUXCONFIG
785               PATH=${PATH}:\$TUXDIR/bin:\$APPDIR; export PATH
786               LANG=$lang; export LANG
787               COLUMNS=200; export COLUMNS
788               ps -eF '%u %p %a' | awk '\$3 ~ \"tlisten\" && \$0 ~ \"$nlsaddr\" {print
    \$2}' | read pid
789               if [ -n \"\$pid\" ]
790                then
791                  echo \"Listener running on $machine: pid = \$pid\"
792                  echo exit 0
793                else
794                  echo \"No Listener running on $machine\"
795                  echo exit 0
796               fi"
797               if [ -z "$uname" ]
798                 then
799                   print "Host $machine not found"
800                   exit 1
801               fi
802               rsh "$uname" -l "$ADMIN" "$prog1" | awk '
803                NR == 1 {line = $0}
804                NR > 1  { print line; line = $0}
805                END { if (sub("^exit ", "", line)) exit line; print line; exit -1} '
806                boucle_status=`expr $boucle_status \| $?`
807             done
808             exit $boucle_status
809           ;;
810           updtlistscript)
811           appname=$1
812           machine=$2
813           set_environ
814           get_tllog
815           get_tuxval > "appname.tux"
816           . ./appname.tux
817           prog="
818           echo \"$tuxdir/bin/tlisten -d $bridge -l $nlsaddr -u $uid -L $tllog\" >  $app
    dir/tlisten.$appname.$machine
819           chmod ug+x $appdir/tlisten.$appname.$machine
820           echo exit \$?"
821           if [ -z "$uname" ]
822             then
823               print "Host $machine not found"
824               exit 1
825           fi
826           rsh "$uname" -l "$ADMIN" "$prog" | awk '
827             NR == 1 {line = $0}
828             NR > 1 { print line; line = $0 }
829             END {if(sub("^exit ","", line)) exit line; print line; exit -1}'
830           exit $?
831           ;;
832         tuxBootEnt)
833               appname=$1; shift
834               cmd="\$TUXDIR/bin/tmboot -y $@"
835               set_environ
836               remote_cmd
837               exit $?
```

```
838             ;;
839         tuxShutEnt)
840             appname=$1; shift
841             cmd="\$TUXDIR/bin/tmshutdown -y"
842             set_environ
843             remote_cmd
844             exit $?
845             ;;
846         tuxBootAllMach)
847             appname=$1; shift
848             cmd="\$TUXDIR/bin/tmboot -y -A  $@"
849             set_environ
850             remote_cmd
851             exit $?
852             ;;
853         tuxShutAllMach)
854             appname=$1; shift
855             cmd="\$TUXDIR/bin/tmshutdown -y -A $@"
856             set_environ
857             remote_cmd
858             exit $?
859             ;;
860         tuxShut)
861             appname=$1; shift
862             cmd="\$TUXDIR/bin/tmshutdown -y $@"
863             set_environ
864             remote_cmd
865             exit $?
866             ;;
867         tuxShutAdmMast)
868             appname=$1; shift
869             cmd="\$TUXDIR/bin/tmshutdown -y -M $@"
870             set_environ
871             remote_cmd
872             exit $?
873             ;;
874         tuxShutSvrSect)
875             appname=$1; shift
876             cmd="\$TUXDIR/bin/tmshutdown -y -S $@"
877             set_environ
878             remote_cmd
879             exit $?
880             ;;
881         tuxBootAdmMast)
882             appname=$1; shift
883             cmd="\$TUXDIR/bin/tmboot -y -M $@"
884             set_environ
885             remote_cmd
886             exit $?
887             ;;
888         tuxBoot)
889             appname=$1; shift
890             cmd="\$TUXDIR/bin/tmboot -y $@"
891             set_environ
892             remote_cmd
893             exit $?
894             ;;
895         tuxShutdown)
896             appname=$2
897             cmd="\$TUXDIR/bin/tmshutdown -y $1"
898             set_environ
899             remote_cmd
900             exit $?
901             ;;
902         tuxBootSvrSct)
903             appname=$1; shift
904             cmd="\$TUXDIR/bin/tmboot -y -S $@"
905             set_environ
906             remote_cmd
907             exit $?
908             ;;
```

```
909         tuxBootBBL)
910                 #echo $*
911               appname=$1; shift
912               cmd="\$TUXDIR/bin/tmboot -y $@"
913               set_environ
914               remote_cmd
915               exit $?
916               ;;
917         tuxShowBooted)
918               appname=$1; shift
919               cmd="(echo psr; echo quit)|\$TUXDIR/bin/tmadmin"
920               set_environ
921               remote_cmd
922               exit $?
923               ;;
924         tuxminIPC)
925               appname=$1; shift
926               cmd="\$TUXDIR/bin/tmboot -y -c $@"
927               set_environ
928               remote_cmd
929               exit $?
930               ;;
931         tuxShutPart)
932                 exit_status=0
933               appname=$1;
934               machine=$2; shift
935               set_environ
936               get_tuxconfig | \
937               sed -e "s/=/ /g" -e 's/"//g' -e 's/\\//' -e "s/\*//"  | awk '
938                     $1 == "APPDIR" && mach_section == 1 && mach_found == 1 {
939                           print "APPDIR " $2 > "appname.tux"
940                           mach_section = 0
941                           mach_found = 0
942                     }
943                     $1 == "TUXCONFIG" && mach_section==1 && mach_found==1 {
944                           print "TUXCONFIG " $2 > "appname.tux"
945                     }
946                     $1 == "MACHINES" {mach_section = 1}
947                     $2 == "LMID" && mach_section == 1 && $3 == machine {
948                           print "MACHINE " $1 > "appname.tux"
949                           mach_found = 1
950                     }
951                     $1 == "TUXDIR" && mach_section==1 && mach_found==1 {
952                           print "TUXDIR " $2 > "appname.tux"
953                     }
954               ' "machine=$machine" "appname=$appname"
955               if [ $? != 0 ]
956               then
957                     exit 1
958               fi
959             appdir=`awk '$1 == "APPDIR" {print $2}' appname.tux`
960             tuxconfig=`awk '$1 == "TUXCONFIG" {print $2}' appname.tux`
961             uname=`awk '$1 == "MACHINE" {print $2}' appname.tux`
962             rootdir=`awk '$1 == "TUXDIR" {print $2}' appname.tux`
963             lang=`sed -e 's/=/ /g' -e 's/;/ /g' $ConfDir/$appname.tux |
964                   awk '$1 == "LANG" {print $2}'`
965             prog1="TUXDIR=$rootdir; export TUXDIR
966                    APPDIR=$appdir; export APPDIR
967                    LIBPATH=${LIBPATH}:$rootdir/lib; export LIBPATH
968                    TUXCONFIG=$tuxconfig; export TUXCONFIG
969                    LANG=$lang; export LANG
970                    PATH=${PATH}:\$TUXDIR/bin:\$APPDIR; export PATH
971                    $TUXDIR/bin/tmshutdown -y -P $@
972                    echo \$? > /tmp/rem$appname.$machine.tux"
973             if [ -z "$uname" ]
974               then
975                  print "Host $machine not found"
976                  exit 1
977             fi
978             rsh $uname -l "$ADMIN" "$prog1"
979             rsh_status=`echo $?`
```

```
980               if [ "$rsh_status" -eq "0" ]
981                 then
982                  status=`rsh $uname -l "$ADMIN" "cat /tmp/rem$appname.$machine.tux"`
983                  rsh $MASTER -l "$ADMIN" "rm /tmp/rem$appname.$machine.tux" 2> /dev/nul
    l
984                  rsh $uname -l "$ADMIN" "rm /tmp/rem$appname.$machine.tux"  2> /dev/nul
    l
985               fi
986               if [ "$status" -ne "0" ]
987                 then
988                    exit_status=`expr $exit_status + 1`
989               fi
990               if [ "$exit_status" -ne "0" -o "$rsh_status" -ne "0" ]
991                 then
992                   exit 1
993               fi
994       ;;
995       loadfshm)
996       appname=$1; machine=$2; shift 2
997       set_environ
998       get_tuxval > "appname.tux"
999       . ./appname.tux
1000      prog="
1001      TUXDIR=$tuxdir; export TUXDIR
1002      ROOTDIR=$tuxdir; export ROOTDIR
1003      LIBPATH=${LIBPATH}:$tuxdir/lib; export LIBPATH
1004      LANG=$lang; export LANG
1005      $tuxdir/bin/loadfiles $@
1006      echo \"\nexit \$?\""
1007      if [ -z "$uname" ]
1008        then
1009          print "Host $machine not found"
1010          exit 1
1011      fi
1012      rsh "$uname" -l "$ADMIN" "$prog" | awk '
1013         NR == 1 {line = $0}
1014         NR > 1 { print line; line = $0 }
1015         END {if(sub("^exit ","", line)) exit line; print line; exit -1}'
1016      ;;
1017      Unloadcf)
1018      appname=$1
1019      set_environ
1020      cmd="\$TUXDIR/bin/tmunloadcf"
1021      if [ $# -eq 2 ]
1022        then
1023         filename=$2
1024         remote_cmd > "$filename"
1025      else
1026         remote_cmd
1027      fi
1028      exit $?
1029      ;;
1030     *)
1031      echo "Command $1 does not exist"
1032      exit 1
1033      ;;
1034  esac
```